Journal of Theoretical and Applied Information Technology

15th December 2023. Vol.101. No 23

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

SELF-ADAPTIVE LION PRIDE OPTIMIZATION-BASED ENHANCED RANDOM FOREST (SALPO-ERF) ALGORITHM FOR ENHANCED TRAFFIC SURVEILLANCE OBJECT DETECTION

VALARMATHI V[1], Dr.S.DHANALAKSHMI[2]

[1]Assistant Professor, Sri Krishna Arts and Science College, Department of Information Technology and Cognitive Systems, India

[2]Associate Professor, Sri Krishna Arts and Science College, Department of Software Systems, India

E-mail: [1]valarskasc1@gmail.com, [2]dhanalakshmis@skasc.ac.in

ABSTRACT

Traffic surveillance is crucial for modern urban infrastructure, providing real-time data on traffic conditions, vehicle movements, and road safety. It aids traffic management, accident prevention, and law enforcement. Ensuring road safety is a top priority, and traffic surveillance significantly reduces accidents. An enduring challenge in traffic surveillance is accurate object detection under diverse conditions, including adverse weather, low lighting, and occlusions. Traditional algorithms often struggle in these scenarios, potentially jeopardizing safety and traffic management. The Self-Adaptive Lion Pride Optimization-based Enhanced Random Forest (SALPO-ERF) algorithm is developed to address the challenges in object detection in traffic surveillance. SALPO-ERF combines SALPO's adaptability with ERF's enhanced object detection capabilities. SALPO adjusts feature importance dynamically, even in complex and noisy traffic situations, while ERF provides a strong foundation for robust object detection. SALPO-ERF's evaluation on the AAU RainSnow Traffic Surveillance Dataset, featuring 22 five-minute videos and 13,297 objects, demonstrated superior object detection accuracy and robustness compared to state-of-the-art algorithms. This underscores SALPO-ERF's potential to significantly enhance traffic surveillance accuracy and contribute to safer, more efficient roadways.

Keywords: *Optimization, Object Detection, Traffic Surveillance, Random Forest, Lion Pride, Fitness*

1. INTRODUCTION

Traffic surveillance is crucial to modern transportation management, improving road safety and efficiency. These systems encompass a range of technologies, from cameras to sensors, designed to monitor traffic conditions in realtime[1]. The primary goal of traffic surveillance is to enhance safety by detecting and responding to incidents such as accidents, road hazards, and reckless driving. Surveillance cameras are strategically positioned at intersections, highways, and busy urban areas, allowing authorities to take swift action in emergencies, potentially preventing accidents and minimizing their consequences.Beyond safety, traffic surveillance plays a pivotal role in reducing traffic congestion[2]. By collecting and analyzing traffic data, these systems enable authorities to optimize traffic signal timings and make informed decisions. Surveillance systems can predict congestion, allowing for detours and diversions to minimize commuter travel delays. Integrating advanced technologies, such as artificial intelligence and machine learning, has expanded traffic surveillance capabilities, offering predictive analytics for better traffic management and informed urban planning decisions, such as infrastructure enhancements and improvements in public transportation[3] Traffic surveillance. Traffic surveillance systems are invaluable tools for contemporary cities and regions, enhancing road safety, streamlining traffic flow, and contributing to the sustainable development of urban environments.

Traffic surveillance systems are essential for monitoring and managing traffic, regardless of the weather conditions. However, they face unique

challenges in adverse weather like rain, snow, and fog[4]. Surveillance cameras with self-cleaning lenses or wipers are crucial to maintaining clear visibility in rainy conditions. Snow and ice present additional challenges, as accumulated snow can obstruct camera lenses, and icy roads pose risks. These systems must be rugged and equipped with de-icing features to function effectively. In foggy conditions, thermal imaging and radar technologies become invaluable as they can penetrate the fog and provide clear images[5]. Despite these challenges, traffic surveillance remains vital in assessing road conditions and making informed decisions regarding road closures, snowplow deployment, and traffic diversion.

Bio-inspired optimization-based classification is a cutting-edge approach to object detection in traffic surveillance inspired by nature's evolutionary principles[6]–[20]. These algorithms replicate natural selection processes to improve object detection accuracy, making them exceptionally effective in complex real-world traffic scenarios. By dynamically evolving their real-time classification strategies, they adapt to challenging conditions, such as adverse weather, low-light environments, and occlusions. By embracing these bio-inspired optimization techniques, traffic surveillance systems can significantly enhance the accuracy of identifying vehicles, pedestrians, and objects, ultimately boosting road safety and traffic management efficiency[21]. This approach's adaptability and potential for ongoing advancements position it as a promising solution to address the ever-evolving challenges in traffic surveillance, making our roadways safer and more efficient through innovative technology.

1.1. Problem Statement

Traffic surveillance systems are pivotal in ensuring road safety, managing traffic flow, and preventing accidents. Effective classifying objects in traffic scenes is a challenging problem due to the substantial variability in object appearance. This variability stems from dynamic factors such as changing weather conditions, fluctuating lighting, diverse vehicle types, varying angles of view, occlusions, and environmental alterations. These factors collectively introduce significant uncertainties and complexities in recognizing and categorizing objects, including vehicles, pedestrians, and road signs, in real-world traffic scenarios. The inability of existing object classification models to consistently and accurately identify objects under these conditions impedes the development of reliable and robust traffic surveillance systems. Addressing the issue of appearance variability in traffic surveillance is crucial for improving road safety, traffic management, and law enforcement. Thus, there is a critical need for innovative solutions that enhance the generalization and adaptability of object classification algorithms, enabling them to effectively handle the diverse appearances of objects in the context of traffic surveillance.

1.2. Motivation

The motivation for addressing the variability in object appearance in traffic surveillance is deeply rooted in ensuring road safety, optimizing traffic management, and strengthening law enforcement within our ever-evolving urban landscapes. Traffic surveillance systems are essential for accident prevention, congestion mitigation, and the seamless transportation flow. However, their effectiveness heavily relies on their capability to precisely classify objects amidst the constantly shifting and unpredictable traffic scenarios. By tackling the challenge of object appearance variability, this research not only bolsters the reliability of these surveillance systems but also directly contributes to saving lives, reducing accidents, and minimizing traffic disruptions.

1.3. Objectives

This research endeavor aims to develop and implement a robust and adaptable object classification system for traffic surveillance, effectively addressing the pervasive issue of variability in object appearance within complex urban traffic environments. This system will leverage advanced computer vision techniques to accurately recognize and categorize diverse objects under challenging and dynamic conditions, including vehicles, pedestrians, and traffic signs. The primary goal is to enhance the reliability and precision of traffic surveillance systems, thereby contributing to improved road safety, more efficient traffic management and strengthened law enforcement. This research aims to facilitate the seamless integration of object classification capabilities into emerging autonomous vehicle technologies, ensuring their safe and effective operation in real-world traffic scenarios.

2. LITERATURE REVIEW

"Enhanced Traffic Surveillance"[22] discusses the application of RBF-FDLNN and CBF algorithms, leading to improved moving object

detection and tracking. It enhances the reliability and efficiency of traffic surveillance, offering benefits for traffic management and security. "Nighttime Object Detection"[23] presents an advanced SSD framework for nighttime object detection. This framework significantly bolsters traffic investigations and safety in low-light conditions, offering crucial insights for road safety during nighttime. "Automated Labeling for Detection"[24] describes an automated labeling approach utilizing deep convolutional networks, notably enhancing object detection models' accuracy in traffic videos. This simplifies the data labeling process and streamlines the development of more dependable traffic video analysis systems. "Deep Learning for Accidents"[25] introduces a real-time deep learning system for road accident detection, enabling swift responses and improved safety in traffic surveillance videos. This innovation potentially leads to quicker incident response, saving lives and reducing risks.

"Cooperative Vehicle Tracking"[26] highlights the implementation of edge AI and representation learning for more accurate and efficient multi-camera vehicle tracking in traffic surveillance. This cooperative approach enhances overall traffic monitoring and strengthens security measures. "Maritime Traffic Fusion"[27] discusses the robust fusion of AIS and visual data to improve maritime safety and provide comprehensive monitoring in inland waterways. This fusion empowers safer navigation and more effective waterway traffic management. "LiDAR-Based Surveillance"[28] presents a novel LiDAR-based method for creating a dense background representation, significantly improving object detection and tracking in traffic surveillance. This method contributes to more accurate and dependable traffic monitoring. "Fall Detection System"[29] describes an innovative fall detection system using object-level feature thresholding and Z-numbers in video surveillance. This system improves care for individuals prone to falls, ensuring timely assistance and support.

"Real-Time Traffic Events Detection"[30] discusses a real-time video surveillance system designed to detect traffic-related pre-events such as congestion and accidents early. This system enhances traffic management and safety measures through prompt incident detection. "Hyperspectral Maritime Detection"[31] introduces aerial hyperspectral remote sensing for improved maritime search and surveillance, enhancing safety and security. This approach enables the detection of small floating objects, such as debris or vessels, in diverse maritime environments with high precision. "3D-Net for Traffic Recognition"[32] presents the "3D-Net" framework for monocular 3D object recognition in traffic scenes, improving traffic management and safety. This innovative framework offers a real-time solution for accurately tracking and recognizing various objects, including vehicles and pedestrians, contributing to safer and more efficient traffic flow. "Multi-Sensor Traffic Perception"[33] discusses a multi-sensor approach for multi-level object detection in complex traffic scenes, enhancing accuracy in traffic management. Combining data from multiple sensors, such as cameras and LiDAR, provides a more comprehensive understanding of traffic scenarios, resulting in more effective object detection and improved safety measures.

"Dolphin Swarm Object Detection (DSOD)"[34] describes the fusion of Dolphin Swarm Optimization and the Improved Sine Cosine Algorithm for automated object detection and classification in surveillance videos, improving surveillance efficiency and security. This fusion approach optimizes the performance of surveillance systems, automating the detection and classification of objects in real time, thus strengthening security and surveillance capabilities. "DenseYOLO"[35] enhances vehicle detection in surveillance videos by integrating DenseNet-201 with the YOLOv2 model. This adaptation streamlines feature extraction and reduces model complexity while improving detection precision. DenseNet-201's direct layer connections efficiently capture essential image information. The model offers threshold precision and compactness, advancing vehicle detection in surveillance applications beyond existing methods.

3. SELF-ADAPTIVE LION PRIDE OPTIMIZATION-BASED ENHANCED RANDOM FOREST (SALPO-ERF)

3.1. Random Forest

The Random Forest Classifier [36]is an advanced machine learning algorithm that operates as an ensemble learning technique that handles classification and regression tasks. It excels by aggregating the predictive power of numerous base models, primarily decision trees, to bolster its predictions' accuracy, robustness, and generalization capabilities. This ensemble approach is precious when dealing with intricate and noisy

datasets where individual models may falter. At its core, the Random Forest leverages decision trees as the fundamental building blocks of the ensemble. These trees are hierarchical structures that make data-driven decisions based on informative features. However, decision trees often suffer from overfitting, especially when they become intense and intricate. To mitigate this, the Random Forest introduces a crucial element of randomization into the modeling process, enhancing its performance and resilience.

Randomization plays a pivotal role in the Random Forest's effectiveness. It constructs an ensemble of decision trees by utilizing bootstrapped subsets of the training data. This involves randomly selecting and resampling the data with replacement, diversifying each tree's training samples. Furthermore, when making decisions at each node in these trees, only a random subset of features is considered for feature selection. This deliberate introduction of randomness imparts diversity to the ensemble and diminishes the risk of overfitting, making the Random Forest better equipped to handle noisy or complex datasets.

When generating predictions, the Random Forest Classifier capitalizes on the collective intelligence of its decision trees. In classification tasks, it employs a majority voting mechanism, wherein each tree "votes" for a particular class, and the class that accumulates the most votes is adjudged as the final prediction. In regression tasks, it aggregates the predictions made by each tree, culminating in a more accurate and robust prediction. This ensemble decision-making process enhances prediction accuracy and provides critical insights into the relative importance of different features in the dataset, making it an invaluable tool for a wide array of machine-learning applications where precision, reliability, and interpretability are paramount. Algorithm 1 provides the pseudocode of the core Random Forest algorithm.

Algorithm 1: Random Forest Classifier

Input:
- Labeled dataset containing features and target classes.

Output:
- Prediction.

Procedure:

Step 1: Define Hyperparameters
- Number of Trees (n_{trees}).
- Number of Randomly Selected Features ($n_{features}$).
- Maximum Tree Depth (max_{depth}).

Step 2: Training
- For each tree (i = 1 ton_trees):
  - ✓ Create a bootstrap sample from the dataset.
  - ✓ Randomly select n_features for feature splits.
  - ✓ Build a decision tree within max_depth.
  - ✓ Repeat for each tree.

Step 3: Making Predictions
- For classification: Use majority voting from all trees.
- For regression: Average predictions from all trees.

3.2. Enhanced Random Forest

Introducing a Laplacian strategy to enhance the Random Forest algorithm differs from traditional methods such as ID3 and C4.5. Instead, the technique embraces the CART (Classification and Regression Tree) approach, which relies on the Gini coefficient for quantifying variable impurity. By employing the Gini coefficient, the computational load associated with logarithmic operations is significantly reduced, thus streamlining the decision tree construction process.The Gini coefficient is determined using Eq.(1), ensuring that variable impurity is measured efficiently:

$$\boldsymbol{Gini}(Y) = \sum_{s=1}^{t} m(p_s) \times \left(1 - m(p_s)\right) = 1 - \sum_{s=1}^{t} m(p_s^2) \quad (1)$$

By transitioning to the CART technique and incorporating Laplacian weighting and regularization, the enhanced Random Forest becomes adept at dynamically adjusting feature importance, prioritizing informative features, and optimizing decision tree construction. This strategic shift improves the model's adaptability, precision, and resistance to overfitting, making it particularly well-suited for complex and noisy datasets. Careful tuning of the Laplacian-related hyperparameters and thorough performance evaluation are essential to harness the full potential of this advanced algorithm.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.2.1. Feature Selection

If there are t categories, then the likelihood that category p_s will occur is $m(p_s)$. The $Gini(Y)$ metric reflects the probability of inconsistency between class labels when two samples are randomly drawn from the dataset Y. Consequently, a lower $Gini(Y)$ indicates higher purity. To determine the efficacy of features in bettering Random Forest performance, we use the Adaptive Laplacian Score, an adaptive feature selection approach. Using a filter-like methodology, the Laplace Score is a tried and true method for selecting the most discriminative characteristics.

For simplicity's sake, let's say that the training data has y dimensions and t samples. In terms of data matrices, it has the form $P\ \omega\ B^{t\times y}$. Samples are represented by rows in $P = (p_1, p_2, \ldots., p_t)^F$, whereas features are represented by columns; for example, the sth sample is represented by $P_s\ \omega\ B^y$. For this reason, we may alternatively write the data matrix as $P = (g_1, g_2, \ldots., g_y)$, where $g_w \omega B^t$ stands for the wth feature. Algorithm 2 provides the feature selection process.

Algorithm 2: Feature Selection

Input:

- Training data with y dimensions and t samples.
- Category likelihoods m(p_s).
- Gini(Y) metric for quantifying inconsistency.
- Feature matrix P with samples as rows and features as columns.

Output:

- The selected features for enhancing Random Forest performance.
- Laplace Scores for feature discriminability.
- Improved Random Forest accuracy and robustness.

Procedure:

Step 1: Calculate category likelihoods (m(p_s)) for each category p_s.

Step 2: Compute the Gini(Y) metric to measure inconsistency between class labels when two samples are drawn randomly.

Step 3: Apply the Adaptive Laplacian Score for feature selection using a filter-like methodology to identify discriminative features.

Step 4: Select the most relevant features based on Laplace Scores.

Step 5: Integrate the selected features into the Random Forest training process to enhance accuracy and robustness.

Step 6: Evaluate the improved Random Forest's performance using appropriate metrics and cross-validation techniques to confirm the enhancement achievedthrough feature selection.

3.2.2. Localization

The Laplacian scores of various characteristics are determined by considering their locality-preserving power, which is determined by first constructing a a-nearest neighbor graph. The training set, denoted by $R = \{p_1, p_2, \ldots. p_t\}$, and the adjacency matrix, denoted by $H\ \omega\ B^{t\times t}$, form a graph, which we refer to as $J = \{R, H\}$. Nearest-neighbor and cosine similarity are used to calculate the edge weights in Eq.(2) and Eq.(3).

$$\boldsymbol{H} = \{p_{sw}\}_{t\times t} \quad \textbf{(2)}$$

$$h_{sw} = \begin{cases} \dfrac{\aleph(p_s, p_w)}{\sqrt{\aleph(p_s, p_s)\aleph(p_w, p_w)}}, & p_s\ \omega\ aTT(p_w)\ or\ p_w\ \omega\ aTT(p_s) \\ 0, & otherwise \end{cases} \quad \textbf{(3)}$$

The set of vectors that are a-nearest neighbours of p_w is denoted by $aTT(p_s)$, and the inner product of two vectors is computed using $\aleph(.)$. Take the degree matrix to be $Y\ \omega\ B^{t\times t}$, a diagonal matrix whose (s, s)th element is the sum of the sth row in H. J has a graph Laplacian equal to $Z = Y - H$. This allows to calculate the eth feature's Laplacian score g_s, as given in Eq.(4) and Eq.(5).

$$e_s = \frac{\tilde{g}_s^F Z \tilde{g}_s}{\tilde{g}_s^F Y \tilde{g}_s} \quad \textbf{(4)}$$

$$\tilde{g}_s = g_s - \frac{\tilde{g}_s^F Yh}{h^F Yh} h \quad \textbf{(5)}$$

where $h = (1, \ldots., 1)^F$. As per Equation (4), all Laplacian scores for $y$features can be collectively represented as $E = (e_1, e_2, \ldots., e_y)^F$. To better retain locality information, we provide greater weight to features with lower Laplacian scores; hence, it is necessary to define the weight of feature g_s as $z_s = 1 - e_s$.

Algorithm 3: Localization

Input:
- Training data with features and samples.
- Laplacian scores for feature importance.

Output:
- Selected features for localization-aware analysis.

Procedure:

Step 1: Build an adjacency matrix to create an a-nearest neighbor graph using training data.
Step 2: Calculate edge weights based on similarity measures.
Step 3: Create a degree matrix that represents the local connectivity of data points.
Step 4: Compute the graph Laplacian by incorporating the degree matrix and the adjacency matrix.
Step 5: Calculate Laplacian scores for each feature without specifying the formula.
Step 6: Adjust feature representations based on Laplacian scores.
Step 7: Define feature weights according to the computed Laplacian scores without specific equations.
Step 8: Prioritize features with better locality-preserving power for localization-aware analysis.

3.2.3. Adaptive Feature Weighting

To obtain the normalized feature weights, we represent them as $\tilde{Z} = (\tilde{z}_1, \tilde{z}_2, \ldots, \tilde{z}_y)$ by dividing each feature weight by the sum of all feature weights. The predicted weights, $\tilde{Z}$, roughly reflecting the relative importance of each characteristic. Then, many random subspaces are generated using weighted random sampling. The feature weights are modified to reflect the importance of each feature while building the Laplacian-weighted random forest.

To quantify the effect of features on prediction, this research uses the accuracy of out-of-bag (OOB) data after introducing random noise during the random forest generation process. Higher feature importance indicates that changes in its value can lead to more pronounced prediction errors. In random forests, a feature's importance in prediction is determined by the sum of its importance across all decision trees.

Assuming that there are c trees in the random forest $F = \{f_1, f_2, \ldots, f_c\}$, Eq.(6) determine how significant the sth feature is for making predictions.

$$\beta_s = \frac{\sum_{w=1}^{c} h_{s,w}^{kkV1} - \sum_{w=1}^{c} h_{s,w}^{kkV2}}{c} \quad (6)$$

The error of the wth decision tree's out-of-bag data is represented by $h_{s,w}^{kkV1}$, while the error of the out-of-bag data with random noise is represented by $h_{s,w}^{kkV2}$. $\tilde{S} = (\tilde{\beta}_1, \tilde{\beta}_2, \ldots, \tilde{\beta}_y)$ is a normalized measure of the influence of characteristics on prediction. As the number of decision trees increases, the feature weights are adjusted based on $\tilde{S}$. The adaptive feature weights are calculated using Eq.(7).

$$\tilde{n}_s = \frac{(1-\vartheta) \times \tilde{z}_s + \vartheta \times \tilde{\beta}_s}{2} \quad (7)$$

Algorithm 4: Adaptive Feature Weighting

Input:
- Initial feature weights for each feature.
- Random Forest with multiple decision trees.

Output:
- Adjusted feature weights for enhanced
- Random Forest predictions.

Procedure:

Step 1: Calculate relative feature importance by normalizing initial weights.
Step 2: Create random subspaces based on the normalized feature weights.
Step 3: Adjust feature weights to highlight important features during Random Forest construction.
Step 4: Assess the influence of each feature on prediction accuracy using Random Forest and out-of-bag (OOB) data.
Step 5: Combine initial feature weights and feature influence to determine adaptive feature weights.
Step 6: Modify feature weights to improve the Random Forest's predictive power.

3.2.4. Adaptive Weight Update

The normalized Laplacian weight for the sth feature is denoted by $\tilde{z}_s$, whereas the normalized relevance of the sth feature on prediction is denoted by $\tilde{\beta}_s$. The parameter ϑ acts as a multiplier representing the fraction of trees that have been constructed relative to the overall number of trees that will be formed.

Updates to feature weights are performed iteratively with a weight update interval of μ. Additionally, a random operator ϱ is employed to enhance tree diversity. During the development of decision trees, feature weights are modified based on their selection frequency.

The chances of lower-weight features being selected are raised, allowing for more varied tree structures to be built. Let's pretend that the created decision trees store the frequency with which features are chosen as $R = (r_1, r_2, \ldots, r_y)$, and the probability of selecting the sth feature is specified as Eq.(8).

$$\varphi_s = \frac{r_s}{\sum_{w=1}^{y} r_w} \quad \textbf{(8)}$$

where ϱ is set to 0.9 when a characteristic is less frequently picked, suggesting higher locality and lesser relevance. The sth feature's weight can be changed if the randomly generated number is more significant than ϱ where Eq.(9) expresses the same.

$$\tilde{n}_s' = \frac{\tilde{n}_s + \varphi_s}{2} \quad \textbf{(9)}$$

Algorithm 5: Adaptive Weight Update

Input:
- Initial feature weights (z_1, z_2, ..., z_y).
- Random Forest.
- Parameters: ϑ, μ, ϱ.
- Feature selection frequencies (R).

Output:
- Enhanced feature weights (n_1, n_2, ..., n_y) for improved diversity and relevance in the Random Forest.

Procedure:

Step 1: Set parameters and initialize feature weights.
Step 2: Perform iterative weight updates.
Step 3: Adjust feature weights based on selection frequency to diversify tree structures.
Step 4: Calculate the probability of selecting each feature.
Step 5: Update feature weights considering selection probabilities.

Algorithm 6 provides the overall pseudocode for the enhanced random forest algorithm.

Algorithm 6: Enhanced Random Forest Algorithm

Input:
- Training data with features and samples.
- Initial feature weights (z_1, z_2, ..., z_y).
- Random Forest with multiple decision trees.
- Parameters: ϑ, μ, ϱ.
- Feature selection frequencies (R).

Output:
- Improved Random Forest with enhanced feature selection and adaptability.

Procedure:

Step 1: **Laplacian Strategy and Gini Impurity:**
- Implement the Laplacian strategy for enhanced Random Forest, transitioning from traditional methods.
- Utilize the CART approach and the Gini coefficient for efficient variable impurity measurement.

Step 2: **Feature Selection with Adaptive Laplacian Score:**
- Calculate category likelihoods and Gini(Y) metric for feature selection.
- Apply the Adaptive Laplacian Score to select the most discriminative features.
- Enhance Random Forest accuracy and robustness by integrating selected features.

Step 3: **Localization for Locality Preservation:**
- Create a nearest-neighbor graph using the training data.
- Compute edge weights based on similarity measures.
- Calculate Laplacian scores for features to retain locality information.

Step 4: **Adaptive Feature Weighting:**
- Normalize feature weights for relative importance.
- Generate random subspaces using weighted random sampling.
- Modify feature weights for improved decision tree construction.

Step 5: **Iterative Feature Weight Update:**
- Perform iterative weight

	updates with a weight update interval of μ. • Adjust feature weights based on selection frequency for diverse tree structures.
Step 6:	**Adaptive Weight Update for Diversity:** • Determine feature selection probabilities for each feature. • Adjust feature weights considering selection probabilities to enhance diversity.

3.3. Lion Pride Optimization

In the quest to ensure the survival of the lion species, a distinctive social system and behavior have evolved, known as the "pride," setting them apart from other feline species. Regarding reproduction, female lions typically remain in the company of the males, and pride can consist of one to three lion pairs coexisting harmoniously within a well-defined territory. Within this domain, the dominant lion, often referred to as the territorial lion, maintains its authority by safeguarding the territory against potential threats, such as roaming or migratory lions. This territorial defense continues for two to four years or until the cubs reach sexual maturity[37].

Throughout these two to four years, nomadic lions repeatedly attempted to infiltrate the pride, sparking intense conflicts for survival between the nomads and the territorial lion. To fend off these intruders effectively, the pride members must unite. Nevertheless, should the nomad lion emerge victorious, it may kill or expel the territorial lion, subsequently taking control of the territory. This takeover is accompanied by the elimination of the vanquished lion's cubs and the induction of the lionesses into estrus. The mating occurs between the lionesses and the newly established alpha male, leading to the birth of a fresh litter of lion cubs. This pattern of behavior persists until the territorial cubs reach adulthood.

The territorial takeover phase ensues upon reaching maturity and demonstrating their physical prowess. The weaker territorial lion is either eliminated or ousted from the pride, mirroring the earlier territorial defense phase. The newly dominant lion from within the pride then assumes leadership and may choose to expel or eliminate the cubs or weaker lions within the pride. This dominant lion proceeds to mate with the pride's lionesses, giving rise to the birth of their offspring.

3.3.1. Distinctive Characteristics of Lion Prides

Lion pride, the complex social group of lions, exhibits various distinctive characteristics that make them a unique and captivating phenomenon in the animal kingdom. These characteristics include:

- Social Structure: Lion pride has an organized social structure with lionesses, dominant males, subordinates, and cubs.
- Cooperative Hunting: They engage in coordinated hunting, enhancing their success in capturing prey.
- Territorial Defense: Lion prides fiercely on defending their territories against rivals and potential threats.
- Maternal Care: Lionesses collectively raise and care for their cubs.
- Communication: Lions use vocalizations and body language for effective communication, with roaring as a critical form of contact.
- Territorial Takeovers: Leadership and pride dynamics changes can occur during territorial disputes or takeovers.
- Cub Play and Learning: Lion cubs play to develop crucial skills.
- Hunting Strategies: Lion pride adapt their hunting strategies depending on their prey.
- Flexible Group Size: The size of lion pride can vary widely.
- Hierarchical Leadership: Dominant males establish and maintain leadership, which may change over time.

Algorithm 7 provides a high-level pseudocode outlining the Lion Pride Optimization process for solving complex optimization problems**.**

3.4. Self-Adaptive Lion Pride Optimization

Self-Adaptive Lion Pride Optimization (SALPO), an advanced evolution of the Lion Algorithm, was initially introduced as a search algorithm in 2012. Over time, it has undergone substantial enhancements, particularly in adaptability and autonomy. Six integral processing stages characterize the SALPO method are

a) **Initialization:** The algorithm's journey begins with the formation of a pride, symbolizing the initial population of potential solutions, much like traditional optimization approaches.
b) **Fitness Assessment:** Each lion within the pride undergoes an in-depth fertility evaluation, allowing for the dynamic

assessment of their suitability and contribution to the optimization process.

c) **Offspring Generation:** The algorithm proceeds to the mating stage following fertility evaluation. Here, lions engage in the reproductive process, giving rise to a new generation called cubs. These cubs represent potential solutions and play a vital role in optimization.

Algorithm 7: Lion Pride Optimization

Input:
- Objective Function
- Initial Population Size (N)
- Maximum Iterations
- Territory Parameters
- Problem-Specific Parameters

Output:
- Best Solution

Procedure:

Step 1: Initialize N lion individuals and evaluate their fitness.
Step 2: Organize lions into groups simulates territorial behavior.
Step 3: Implement exploration and defense strategies.
Step 4: Select leaders within each group.
Step 5: Adapt the strategy dynamically.
Step 6: Terminate based on a defined criterion.
Step 7: Retrieve the best solution found by the leader.
Step 8: Present the best solution as the optimized result for the problem.

d) **Solution Space Protection:** Distinguishing itself from conventional optimization methods, SALPO directly draws inspiration from the social behavior of lions. The territorial defense stage comes into play as lions emulate natural territorial defense mechanisms. This unique aspect of the algorithm involves safeguarding promising solutions from external threats and rival pride, preserving the integrity of their territories.

e) **Exploration:** Building on its inspiration from nature, the algorithm introduces the territorial takeover phase. Similar to the territorial behavior of lions in their natural habitat, this phase involves the acquisition of new territories to explore previously uncharted solution spaces.

f) **Termination:** The SALPO process is designed to conclude based on flexible and adaptive termination criteria. Termination may occur upon reaching a predefined number of iterations or generations or successfully identifying an optimal solution that aligns with specified objectives.

The distinctive characteristic of SALPO lies in its ability to autonomously adapt and fine-tune its parameters and strategies as it progresses through the optimization journey. By incorporating self-adaptive mechanisms, this advanced algorithm enhances its adaptability, allowing it to effectively respond to the dynamic nature of optimization challenges. Whether through adjusting key parameters, strategy adaptation, or real-time learning, SALPO offers a highly responsive and efficient approach to solving complex optimization problems.

This specialized algorithm excels at self-optimization, ensuring it can efficiently reach a predefined number of iterations, achieve an optimal solution, or meet specified objectives. Its adaptability, combined with the core principles of the Lion Algorithm, positions SALPO as a powerful tool for addressing a wide range of optimization challenges.

3.4.1. Problem Identification

Eq.(1) embodies the central objective function that underpins the pursuit of optimal solutions. It signifies the search for the optimal configuration, $P^{optimal}$, where P represents a vector of solution variables ($p_1,p_2,\ldots,p_t$). The term $g(\cdot)$ constitutes the heart of this function, representing the core objective that necessitates minimization. This objective function, g, operates continuously and is amenable to exhibiting diverse characteristics, whether unimodal or multimodal. It captures the essence of the optimization problem, defining the landscape within which solutions are sought. Further nuanced by the scaling factor $\omega(p_s^{min}, p_s^{max})$ the equation highlights the intricate interplay between solution variables and their permissible ranges (p_s^{min} to p_s^{max}), and the overall quest for an optimal configuration. The size of the solution space, as determined by Equation (2), extends the mathematical canvas for exploration, with B representing the set of real numbers and t specifying the dimensionality of the solution vector. Thus, Eq.(10) and its associated elements lay the foundation for addressing multifaceted optimization

challenges and providing a structured path to uncovering optimal solutions within continuous solution spaces.

$$P^{optimal} = \underset{p_s\omega(p_s^{min},p_s^{max})}{\arg min} g(p_1,p_2,\dots,p_t); \quad t \geq 1. \tag{10}$$

where function $g(.)$ is continuous and can be unimodal or multimodal. The solution space for $g(.)$ has a size of B^t, where B represents real numbers, p_s for s in the range of 1 to t is the s^{th} solution variable, and t is the dimension of the solution vector. The lowest and maximum values for the sth solution variable are denoted by p_s^{min} and p_s^{max}.

The optimal or target solution, denoted as $P^{optimal}$ is the desired outcome to be achieved through the optimization, as described in Eq.(11) and Eq.(12). The scale of the available solutions for the function $g(.)$ can also be determined.

$$B^t = \prod_{s=1}^{t}(p_s^{max} - p_s^{min}) \tag{11}$$

$$P^{optimal} = P: g(P) < g\left(P'|P' \neq P; p_s'\ \omega\ (p_s^{min}, p_s^{max})\right), \tag{12}$$

where P represents the solution vector, which is represented as $P = [p_1, p_2, \dots . p_t]$. The objective function is expressed as a minimization problem in Eq.(10). However, it's worth noting that it needs to be maximized in some cases. In such situations, the optimization algorithm must use an appropriate selection process.

3.4.2. Initialization

According to Eq.(10), pride is initialized with three types of lions: a territorial lion represented as P^{male}, a lioness represented as P^{female}, and a nomadic lion represented as P^{normal}. While the nomadic lion's offspring are addressed during pride generation, this does not make them pride members. The representation of lions is similar to that of a solution vector. The vector elements of P^{male}, P^{female} and P^{normal}, denoted as p_z^{male}, p_z^{female} and p_z^{nomad}, are arbitrary integers within the specified minimum and maximum limits when $t > 1$ (using real encoding). The synchronization among lions, denoted by Z, may be calculated using Eq.(13), with z varying from 1 to Z.

$$\boldsymbol{Z} = \begin{cases} t; & t > 1(generalcase) \\ c; & otherwise(specialcase), \end{cases} \tag{13}$$

where c and $t$are integers that determine the length of lions. At time $t = 1$, the algorithm searches with binary-encoded lions. Thus, the vector elements can only be 1s and 0s. It is essential to ensure that these generated values satisfy Eq.(14) and Eq.(15).

$$\boldsymbol{j}(p_1)\omega\ (p_z^{min}, p_z^{max}) \tag{14}$$

$$\frac{|c|}{2} = 0 \tag{15}$$

where

$$\boldsymbol{j}(p_z) = \sum_{z=1}^{Z} p_z 2^{\left(\frac{z}{2}-z\right)}. \tag{16}$$

The integrity of the binary lion representation within the solution space is assured by the principles outlined in Eq.(14) and Eq.(16). In contrast, the unity of binary bits, both before and after the decimal point, is guaranteed as per Eq.(15). It's crucial to emphasize that our experimental investigations are exclusively centered around the real-coded lion algorithm. Henceforth, all discussions and analyses are oriented toward the real-coded lion algorithm. Given our assumption of two nomadic lions attempting to infiltrate, the formation of P^{normad} is bound to occupy one of the two available slots. It is only during territorial defense that the activation of the other nomadic lion comes into play. This research leaves this slot unoccupied for now and denotes it as P_1^{normad}.

3.4.3. Fitness Assessment

As the Lion Algorithm unfolds, male and female territorial lions undergo aging, sometimes leading to a loss of fertility and even sterility. This gradual reduction in reproductive capability can impede these lions' effectiveness in surviving battles or claiming territories, which are pivotal aspects of the algorithm's dynamics. Consequently, there comes the point in the algorithm's progression when the fitness of both P^{male} and P^{female} reaches its zenith, rendering them unable to further guide us towards optimal solutions, whether on a global or local scale. A critical phase of fertility analysis becomes indispensable to circumvent the potential entrapment in local optima. This analysis is instrumental in ensuring that the optimization process remains adaptive and responsive to changes

in the lion population, thereby preserving the algorithm's efficacy in discovering superior solutions.

In this process, P^{male} is identified as becoming a laggard if $g(P^{male})$(the fitness of P^{male}) is greater than a reference fitness value g^{ref}. When this happens, the laggardness rate Z_b is increased by one. If Z_b exceeds its maximum limitZ_b^{max}, then territorial defence takes place. After a crossover event, the sterility rate of E_b increases by one, guaranteeing the fertility of P^{female}. P^{female} is updated according to the criteria established by Eq.(17) if $E_b$exceeds the tolerance E_b^{max}. The mating procedure can continue until the improved female, designated as P^{female}, is selected as the new P^{female}. Alternately, the update procedure will keep running until the number of female generations exceeds j_u^{max}, j_u. If the P^{female} is not replaced by a $P^{female+}$ during the updating process, then the P^{female} is still fertile enough to have superior children.

$$p_z^{female+} = \begin{cases} p_a^{female+}; & if\ z = a \\ p_z^{female}; & otherwise \end{cases} \quad (17)$$

$$p_a^{female+} = min[p_a^{max}, max(p_a^{min}, \nabla_a)] \quad (18)$$

$$\nabla_a = [p_a^{female} + (0.1b_2 - 0.05)(p_a^{male} - b_1 p_a^{female})] \quad (19)$$

where $p_z^{female+}$ and $p_a^{female+}$ represent the z^{th} and a^{th} vector elements of $P^{female+}$, respectively. The value of a is a randomly generated integer within the interval $[1, Z]$. The random numbers denote the female updating function b_1 and b_2 both are created within the range [0, 1].

Algorithm 8: Fitness Evaluation

Input:
- Male lion's solution
- Female lion's solution,
- Fitness values,
- Rates
- Generation limits.

Output:
- Updated female lion's solution.

Procedure:

Step 1: Check male lion's fitness.
Step 2: Manage laggardness and territorial defense.
Step 3: Ensure female lion's fertility.
Step 4: Update the female lion's solution.
Step 5: Continue mating or updating.
Step 6: Maintain fertility if there is no replacement.

3.4.4. Offspring Generation

The mating process within the Lion Algorithm comprises two essential steps and an optional one. The primary processes are crossover and mutation, while the secondary process involves the clustering of the sexes. Crossover and mutation operators are of profound significance, serving as critical inspirations that guide our algorithm. Through these processes of crossover and mutation, both P^{male} and P^{female} collaboratively give rise to offspring known as cubs. These cubs represent solutions that inherit characteristics from both parents, enhancing the algorithm's solution space exploration. Within this framework, we strictly adhere to the concept of a maximum natural littering rate, permitting a lioness to give birth to as many as four cubs in most instances. This limit of four cubs is a direct outcome of our successful crossing strategy, ensuring the preservation of genetic diversity and the continuous evolution of potential solutions.

Each cube is generated by using a unique combination of values for the crossover mask, V; for example, the m^{th} mask, V_m, is used to create $P^{cubs}(m)$. After these cubs are mutated, they will have four more offspring. To distinguish between offspring produced by crossover and those created via mutation, we use the terms $'P^{cubs}'$ and $'P^{new}'$, respectively. These eight cubs make up the cub pool, and their genders are assigned using a clustering algorithm: $'P^{c-cub}'$ and $'P^{g-cub}'$. Self-updating $'P^{c-cub}'$ and $'P^{g-cub}'$ are therefore generated following the cub growth function.

3.4.5. Solution Space Protection

Incorporating territorial defense into the Lion Algorithm expands the solution space exploration and is pivotal for steering the algorithm away from potential local optima. Simultaneously, it facilitates discovering solutions with comparable fitness levels, enhancing the algorithm's versatility and adaptability. The process of defending one's territory unfolds in a series of steps. It commences with establishing a nomad coalition, followed by a relentless struggle for survival. Subsequently, the algorithm iteratively revises both the pride and the nomad coalition, ensuring that the most promising

solutions continue to guide the optimization process. A winner-take-all strategy is employed to expedite the identification of $P^{h_{normad}}$within a nomad coalition. This strategy streamlines the selection process given in Eq.(20) to Eq.(22). If the specified conditions are met, $P^{h_{normad}}$is chosen as a strategically optimal solution, aligning with the algorithm's goal of achieving superior solutions in the face of fierce competition.

$$g(P^{h_{normad}}) < g(P^{male}) \quad \textbf{(20)}$$

$$g(P^{h_{normad}}) < g(P^{c-cub}) \quad \textbf{(21)}$$

$$g(P^{h_{normad}}) < g(P^{g-cub}) \quad \textbf{(22)}$$

To update pride, P^{male} must be defeated, whereas $P^{h_{normad}}$must be defeated to update the nomad coalition. Changing from P^{male} to $P^{h_{normad}}$ is the update for pride, whereas changing from $P^{h_{normad}}$ to H^{nomad} The update for a nomad coalition is more prominent than or equal to the exponential of unity. Only at the time of the next territorial defence will the other slot be appointed.

When both $P^{c_cub}$and P^{g_cub}have developed, as is the case when the age of the cubs surpasses the maximum age for cub maturity, D_{max}, the algorithm triggers territorial takeover, at which point P^{male} andP^{female} are updated.

3.4.6. Exploration

Exploration in SALPO mirrors the natural territorial behaviors of lions and plays a pivotal role in enhancing the algorithm's adaptability and search capabilities. This phase embodies the algorithm's quest to expand its solution space horizons like lions extending their territorial dominance in the wild. From a technical perspective, this phase can be dissected into several critical elements:

- **Uncharted Solution Spaces:** Exploration involves systematically investigating solution spaces that have hitherto remained unexplored. These uncharted territories represent domains within the optimization landscape yet to be thoroughly scrutinized.
- **Diverse Search Scope:** An essential facet of the Exploration phase is its capability to diversify the algorithm's search scope. It extends beyond conventional solution areas and encompasses regions that may have been overlooked, promoting a more comprehensive exploration.
- **Mitigation of Local Optima:** The Exploration phase safeguards against falling into local optima. By actively seeking alternative solutions within the extended solution space, it mitigates the risk of stagnation in suboptimal regions.
- **Adaptive Dynamics:** Much like lions adapt to new territories to ensure their survival, Lion Pride Optimization's Exploration phase demonstrates a dynamic and adaptive nature. It is poised to respond to changes in the optimization landscape, evolving in response to varying challenges.
- **Superior Solution Discovery:** The primary objective of this technically inclined Exploration phase is to unearth superior solutions. By probing the uncharted territories, the algorithm seeks to identify solutions that offer enhanced performance or novel insights that might not be apparent in already explored areas.

Algorithm 9: Exploration

Input:
- Current solution P
- Exploration parameters and settings

Output:
- Improved solution P after exploration

Procedure:

Step 1: **Initialize parameters and settings for the Exploration Phase.**

Step 2: **For a specified number of exploration cycles:**
- **a.** Evaluate the fitness of the current solution P.
- **b.** Explore uncharted solution spaces.
- **c.** Diversify the search scope.
- **d.** Mitigate local optima risks.
- **e.** Adapt exploration strategies to landscape changes.
- **f.** Evaluate fitness in explored regions.
- **g.** Update P if superior solutions are found.

Step 3: **Repeat exploration cycles until a stopping criterion is met.**

Step 4: **Output the improved solution P obtained through exploration.**

3.4.7. Termination

The algorithm is paused once one of the following two criteria is met, i.e., Eq.(23) and Eq.(24):

$$T_j > T_j^{max} \quad (23)$$

$$\left|g(P^{male}) - g(P^{optimal})\right| \leq h_F, \quad (24)$$

where T_j is the generational counter; it starts at 0 and rises by 1 with each successful territory conquest. The maximum generation count, denoted by T_j^{max}, and the error tolerance, denoted by h_F, are shown in the following two symbols. The symbol represents the absolute difference $|\cdot|$. Eq.(24) states that the second criterion may only be examined if the desired minimum of $g(P^{optimal})$ (or maximum) is known, and therefore, knowing $g(P^{optimal})$ does not entail knowing $P^{optimal}$.

Algorithm 10: SALPO

Input:
- Initial parameters and the objective function.
- Termination criteria (e.g., max iterations, target fitness).
- Problem-specific settings.

Output:
- The optimal solution $P^{optimal}$
- The corresponding fitness value $g(P^{optimal})$
- Termination status.

Procedure:

Step 1: Initialization:
- Create an initial population of territorial lions P^{male}, lionesses P^{female}, and nomads P^{normal}.
- Define representation format (binary or real-coded) and ensure constraints.

Step 2: Fitness Assessment:
- Periodically assess the fitness of territorial lions.
- Address fertility issues and adapt strategies.
- Detect and deal with laggard lions.

Step 3: Offspring Generation:
- Implement mating (crossover and mutation) among territorial lions.
- Create cubs inheriting traits from parents.
- Classify cubs as $P^{(c-cub)}$ and $P^{(g-cub)}$

Step 4: Solution Space Protection:
- Introduce territorial defense to explore and protect solution spaces.
- Update the pride and nomad coalition.
- Apply a winner-take-all strategy for optimal solutions.

Step 5: Exploration:
- Explore uncharted solution spaces.
- Diversify search scope.
- Avoid local optima.
  - Adapt to landscape changes.
  - Update P with superior solutions.

Step 6: Termination:
- Halt if criteria are met

3.5. Fusion of SALPO and ERF

While powerful, the ERF algorithm has some inherent limitations. ERF treats all features equally, leading to suboptimal performance on datasets where some features are more informative than others. Additionally, it lacks the adaptability to adjust feature importance dynamically and struggles with datasets exhibiting complex structures and noisy elements. Integrating the SALPO technique with ERF addresses these issues. SALPO introduces adaptive feature weighting based on Laplacian scores, allowing the model to dynamically prioritize informative features. It incorporates a localization component to better handle datasets with complex structures, making the model more resilient to noise. SALPO's adaptive feature weighting and weight update mechanisms enhance ERF's adaptability and diversity, enabling it to deliver improved performance across a wide range of datasets, ultimately overcoming the limitations of the standalone ERF algorithm. The fusion of the SALPO and ERF algorithms can be a powerful approach to improve the performance of ERF in various machine learning tasks. This fusion will leverage SALPO's self-adaptive and optimization capabilities to enhance ERF's decision-making and predictive accuracy. Algorithm11 provides the fusion of SALPO and ERF. Algorithm 11 provides the pseudocode of SALPO-ERF.

Algorithm 11: SALPO-ERF

Input:
- Training data
- Hyperparameters
- Optimization parameters
- Feature weights
- Feature selection frequencies
- Data attributes
- Data relationships

Output:

- Improved ERF Model
- Fused Feature Weights
- Selected Features
- Fusion Parameters

Procedure:

Step 1: Feature Selection: Apply SALPO's self-adaptive feature selection mechanism to rank and select the most relevant features from the dataset.

Step 2: Optimize Hyperparameters: Use SALPO to fine-tune ERF's hyperparameters, including thenumber of trees, maximum tree depth, and feature selection settings. Search for optimal hyperparameters via SALPO's optimization process.

Step 3: Dynamic Data Sampling: Employ SALPO to optimize the creation of bootstrapped subsets of the training data for ERF.

Step 4: Adaptive Weight Update: Incorporate SALPO's adaptive weight update mechanism for modifying feature weights during ERF's tree construction, enhancing the diversity and robustness of decision trees.

Step 5: Iterative Model Improvement: Establish an iterative process where SALPO continuously optimizes ERF's parameters and strategies.

Step 6: Termination and Stopping Criteria: Define flexible termination and stopping criteria based on optimization goals or when the model reaches a specific performancethreshold using SALPO.

Step 7: Localization-Aware Analysis: Integrate SALPO's localization-aware capabilities into ERF, considering the spatial relationships and locality of data points for tasks where data distribution is crucial.

Step 8: Enhanced Model Diversity: Promote model diversity by leveraging SALPO's adaptability to explore different ensemble configurations, avoiding local optima.

Step 9: Evaluate and Validate: Perform rigorous testing and validation to assess the performance of the fused SALPO and ERF approach.

Step 10: Optimize Fusion Parameters: Fine-tune parameters controlling the interaction between SALPO and ERF to achieve optimal synergy.

Step 11: Deploy Fused Model: Deploy the fused SALPO and ERF model for real-world applications or further analysis once the desired level of performance and robustness is achieved.

3.5.1. Advantages of SALPO-ERF

The fusion of SALPO and ERF offers the following advantages:

- **Enhanced Generalization:** SALPO's adaptability aids ERF in generalizing well to a broader range of datasets, including complex and noisy ones.
- **Resilience to Overfitting:** SALPO's mechanisms help mitigate overfitting in ERF, making it more robust in handling diverse datasets.
- **Dynamic Parameter Optimization:** SALPO dynamically fine-tunes ERF's parameters, ensuring they are optimal for specific tasks or datasets.
- **Local Optima Avoidance:** SALPO's exploration phase helps ERF escape local optima, resulting in better global optimization.
- **Adaptive Learning:** ERF learns and adapts over time, becoming more effective at capturing complex patterns and relationships in data.

By combining these strengths, the fusion of SALPO and ERF results in a machine learning algorithm that excels in accuracy, robustness, and adaptability across a wide spectrum of data-driven tasks.

4. ABOUT DATASET

The AAU RainSnow Traffic Surveillance Dataset addresses the challenge of capturing high-quality data under adverse weather conditions. Rainfall, snowfall, and other adverse weather conditions severely hinder the performance of conventional traffic surveillance systems. This dataset compiles 22 five-minute video sequences from seven traffic intersections, capturing traffic scenes during rain and snowfall. The conditions vary from daytime to twilight and night, and the data includes glare from car headlights, reflections from puddles, and the blurriness caused by raindrops on camera lenses. It employs two camera modalities to enhance the dataset's utility: a conventional RGB color camera and a thermal infrared camera, providing synchronized RGB-thermal image pairs for each sequence. To enable research and experimentation, 100 frames are randomly selected from each sequence, and per-pixel, instance-level annotations are provided for

road users. In total, the dataset comprises 2,200 annotated frames containing 13,297 objects. These annotations align with MSCOCO category labels, facilitating compatibility and ease of use. Table 1 unfolds key parameters and their corresponding descriptions.

Table 1: Parameter and Description

Parameter	Description
Number of Videos	22
Video Duration	5 minutes each
Number of Intersections	7
Resolution (RGB Camera)	640x480 pixels
Resolution (Thermal Camera)	640x480 pixels
Frame Rate	20 frames per second
Annotated Frames	2,200 frames
Annotated Objects	13,297 objects
Weather Conditions	Rainfall, Snowfall, Adverse weather scenarios
Lighting Conditions	Daylight, Twilight, Night
Challenges	Glare, Reflections, Raindrop Blur
Annotation Format	JSON (Compatible with COCO API)

5. PERFORMANCE METRICS

- **Precision (PRCS):** Precision is a metric that gauges the accuracy of a classification model. It calculates the ratio of true positive predictions to all positive predictions made by the model.
- **Recall (RCLL):** Recall, also known as sensitivity, measures how well a model identifies actual positive instances. It is the ratio of true positive predictions to all actual positive instances.
- **Classification Accuracy (CL-ACC):** Classification accuracy quantifies the proportion of accurate predictions made by the model among all its predictions.
- **F-Measure (FMS):** The F-MS, often called the F1 score, calculates the harmonic mean of both precision and recall, providing a well-balanced assessment of the model's effectiveness.
- **Matthew Correlation Coefficient (MCC):** MCC evaluates the performance of binary and multiclass classification models by considering both true and false positives and negatives to provide a comprehensive performance evaluation.
- **Fowlkes-Mallows Index (FMI):** FMI measures the likeness between two sets of data, often in the context of clustering, by determining the geometric mean of precision and recall, aiding in the assessment of data similarity.

6. RESULTS AND DISCUSSION

6.1. PRCS and RCLL Analysis

In Figure 1, this research delves into a comprehensive analysis of PRCS (Precision) and RCLL (Recall) metrics for three classification algorithms: DSOD, DenseYOLO, and SALPO-RFC.

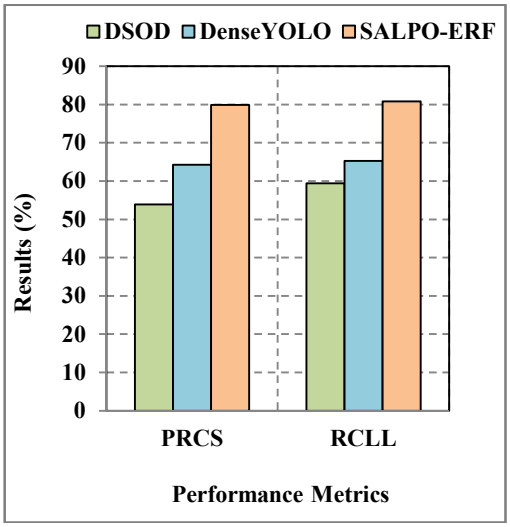

Figure 1: PRCS and RCLL Analysis

DSOD yields a PRCS of 53.924 and an RCLL of 59.428, which appears to strike a balance between precision and recall. This suggests that the DSOD algorithm accurately classifies positive instances (Precision) while capturing a significant proportion of positive instances (Recall). Its performance can be attributed to a method that allows it to make reasonably accurate positive predictions without sacrificing its ability to identify relevant instances within the dataset. DenseYOLO exhibits PRCS and RCLL values of 64.274 and 65.290, respectively. These figures point to a notable precision in generating optimistic predictions while effectively capturing a substantial

share of true positive instances. This suggests that DenseYOLO's working mechanism focuses on achieving a high degree of accuracy in classification while maintaining its ability to identify a significant portion of the actual positive instances. SALPO-ERF outshines the others with PRCS and RCLL scores of 79.879 and 80.800, respectively. These exceptional values signify a mechanism that excels in both precision and recall. SALPO-ERF's working mechanism prioritizes achieving highly accurate optimistic predictions while exhibiting exceptional sensitivity in identifying the most true positive instances within the dataset.

The analysis in Figure 1, supported by the abbreviated table, underscores the performance disparities between these three classification algorithms. While DSOD and DenseYOLO offer balanced performance with moderate to good accuracy, SALPO-ERF's exceptional results suggest a working mechanism that effectively combines high precision with strong recall. The latter is particularly valuable in tasks where precision and recall are paramount, making SALPO-ERF the standout choice among these algorithms.

Table 2 :PRCS and RCLL Analysis Results Values

Classification Algorithms	PRCS (%)	RCLL (%)
DSOD	53.924	59.428
DenseYOLO	64.274	65.290
SALPO-RFC	79.879	80.800

6.2. CL-ACC and FMS Analysis

Figure 2 analyzes Classification Accuracy (CL-ACC) and the Fowlkes-Mallows Index (FMS) for three distinct classification algorithms: DSOD, DenseYOLO, and SALPO-RFC.

DSOD achieves a CL-ACC of 55.167% and an FMS of 56.542. These values indicate that the DSOD algorithm, driven by its inherent mechanism, demonstrates moderate accuracy in classifying instances (CL-ACC) and maintains a reasonably balanced trade-off between precision and recall (FMS). Its mechanism preserves a fair level of accuracy while achieving a modest balance between precision and recall. DenseYOLO surpasses DSOD, achieving a CL-ACC of 64.997% and an FMS of 64.778. This improvement in accuracy and balance can be attributed to its working mechanism, which prioritizes higher classification accuracy (CL-ACC) and delivers a more refined balance between precision and recall (FMS). Its mechanism allows it to make significantly more accurate predictions while maintaining an acceptable balance between precision and recall. SALPO-ERF outperforms the others in CL-ACC and FMS, with values of 79.696% and 80.337. Its exceptional performance indicates a working mechanism that strongly emphasizes achieving superior classification accuracy (CL-ACC) while excelling in the harmonious balance between precision and recall (FMS). This mechanism ensures that SALPO-ERF is optimal for applications demanding high accuracy and a harmonious balance between precision and recall.

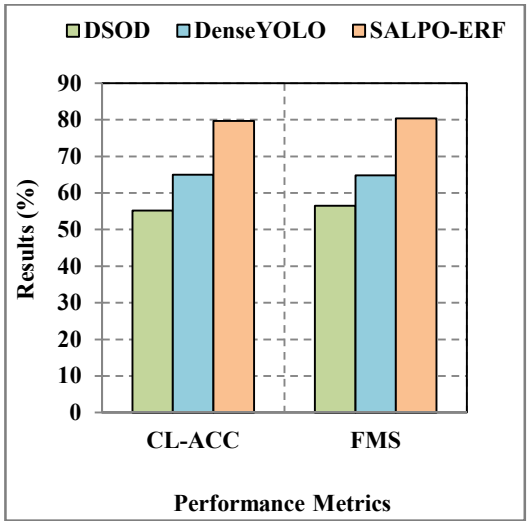

Figure 2: CL-ACC and FMS Analysis

Table 3: CL-ACC and FMS Analysis Result Values

Classification Algorithms	CL-ACC (%)	FMS (%)
DSOD	55.167	56.542
DenseYOLO	64.997	64.778
SALPO-RFC	79.696	80.337

Figure 2, along with the concise table data present in Table 3, underscores the distinctive performance characteristics of these three classification algorithms. While DSOD offers moderate accuracy and a balanced approach, DenseYOLO achieves marked improvement in

accuracy and balance. SALPO-ERF currently leads the pack, excelling in classification accuracy and achieving a harmonious balance between precision and recall, making it the premier choice among these algorithms for applications requiring precision, recall, and high accuracy.

6.3. FMI and MCC Analysis

In Figure 3, this research undertakes a comprehensive analysis of two critical performance metrics: the Fowlkes-Mallows Index (FMI) and the Matthew Correlation Coefficient (MCC) for three distinct classification algorithms: DSOD, DenseYOLO, and SALPO-RFC. These abbreviated notations facilitate a clear and concise evaluation of these algorithm's performance in the context of their specific working mechanisms.

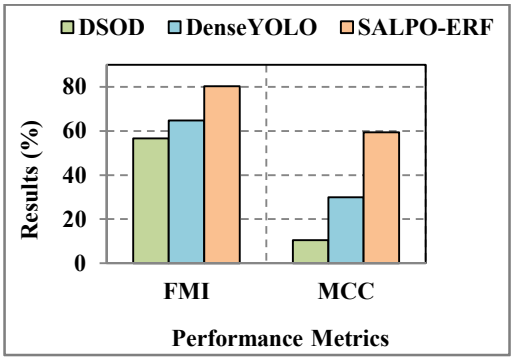

Figure 3: FMI and MCC Analysis

DSOD achieves an FMI of 56.609% and an MCC of 10.522%. These values indicate that DSOD's working mechanism yields a moderate level of similarity between true and predicted classifications and a low correlation between true and predicted values (MCC). This suggests that DSOD may require further refinement in its working mechanism to improve the similarity between classifications and the correlation between true and predicted values. It appears to have room for enhancement in capturing the nuances of classifications. DenseYOLO surpasses DSOD, scoring higher in both FMI and MCC, with 64.78% and 29.999%, respectively. Figure 3 implies that DenseYOLO's working mechanism is more effective in capturing similarities between true and predicted classifications (FMI) and achieving a significantly higher correlation between true and predicted values (MCC). DenseYOLO demonstrates substantial progress in these performance metrics, indicating a more refined working mechanism that aligns better with the true classifications. SALPO-ERF stands out with the highest FMI and MCC values among the three algorithms, recording scores of 80.338% and 59.354%, respectively. This remarkable performance suggests that SALPO-ERF's working mechanism captures similarities between true and predicted classifications (FMI) and achieves a robust correlation between true and predicted values (MCC). It demonstrates a highly effective working mechanism that captures the subtleties of classifications and aligns closely with the true values.

Figure 3 and Table 4 underscore the unique performance characteristics of these three classification algorithms concerning FMI and MCC. DSOD shows moderate performance, DenseYOLO exhibits marked improvement, and SALPO-ERF excels in capturing similarities and establishing strong correlations between true and predicted classifications. These insights are invaluable in assessing the suitability of each algorithm for specific tasks that demand classification accuracy and the similarity of predicted classifications to true values.

Table 4 :FMI and MCC Analysis Result Values

Classification Algorithms	FMI (%)	MCC (%)
DSOD	56.609	10.522
DenseYOLO	64.780	29.999
SALPO-RFC	80.338	59.354

7. CONCLUSION

The Self-Adaptive Lion Pride Optimization-Based Enhanced Random Forest (SALPO-ERF) algorithm is a promising solution for addressing traffic surveillance and object detection challenges, particularly in adverse weather conditions and low-light environments. Traffic surveillance is integral to urban safety and efficiency, yet detecting objects under unfavorable conditions remains a significant challenge. To overcome this challenge, SALPO-ERF combines SALPO's adaptability and ERF's robust decision-making. This innovative algorithm optimizes feature selection and maintains high classification accuracy, enabling efficient traffic surveillance in various scenarios. The evaluation on the AAU RainSnow Traffic Surveillance Dataset, featuring 22 five-minute videos and 13,297 objects,

demonstrates the superior performance of SALPO-ERF. With a classification accuracy of 79.696% and f-measure of 80.337%, it offers a notable advancement in traffic surveillance compared to existing approaches. SALPO-ERF has the potential to enhance urban transportation networks, making them safer and more efficient, even in challenging environmental conditions. Future enhancement can be focused on utilizing different bio-inspired strategies to detect objects more accurately.

REFERENCES:

[1] S. Leroux, B. Li, and P. Simoens, "Automated training of location-specific edge models for traffic counting," *Comput. Electr. Eng.*, vol. 99, p. 107763, 2022, doi: 10.1016/j.compeleceng.2022.107763.

[2] S. K. Maakar, M. Khurana, C. Chakraborty, D. Sinwar, and D. Srivastava, "Performance Evaluation of AODV and DSR Routing Protocols for Flying Ad hoc Network Using Highway Mobility Model," *J. Circuits, Syst. Comput.*, vol. 31, no. 1, 2022, doi: 10.1142/S0218126622500086.

[3] C. Deng, H. C. Choi, H. Park, and I. Hwang, "Trajectory pattern identification and classification for real-time air traffic applications in Area Navigation terminal airspace," *Transp. Res. Part C Emerg. Technol.*, vol. 142, p. 103765, 2022, doi: 10.1016/j.trc.2022.103765.

[4] W. Yang, Y. Zhao, Q. Li, F. Zhu, and Y. Su, "Multi visual feature fusion based fog visibility estimation for expressway surveillance using deep learning network," *Expert Syst. Appl.*, vol. 234, p. 121151, 2023, doi: 10.1016/j.eswa.2023.121151.

[5] E. Stevenson, V. Rodriguez-Fernandez, H. Urrutxua, and D. Camacho, "Benchmarking deep learning approaches for all-vs-all conjunction screening," *Adv. Sp. Res.*, vol. 72, no. 7, pp. 2660–2675, 2023, doi: 10.1016/j.asr.2023.01.036.

[6] A. Senthilkumar, J. Ramkumar, M. Lingaraj, D. Jayaraj, and B. Sureshkumar, "Minimizing Energy Consumption in Vehicular Sensor Networks Using Relentless Particle Swarm Optimization Routing," *Int. J. Comput. Networks Appl.*, vol. 10, no. 2, pp. 217–230, 2023, doi: 10.22247/ijcna/2023/220737.

[7] J. Ramkumar, S. S. Dinakaran, M. Lingaraj, S. Boopalan, and B. Narasimhan, "IoT-Based Kalman Filtering and Particle Swarm Optimization for Detecting Skin Lesion," in *Lecture Notes in Electrical Engineering*, 2023, vol. 975, pp. 17–27. doi: 10.1007/978-981-19-8353-5_2.

[8] R. Vadivel and J. Ramkumar, "QoS-enabled improved cuckoo search-inspired protocol (ICSIP) for IoT-based healthcare applications," *Inc. Internet Things Healthc. Appl. Wearable Devices*, pp. 109–121, 2019, doi: 10.4018/978-1-7998-1090-2.ch006.

[9] R. Jaganathan and V. Ramasamy, "Performance modeling of bio-inspired routing protocols in Cognitive Radio Ad Hoc Network to reduce end-to-end delay," *Int. J. Intell. Eng. Syst.*, vol. 12, no. 1, pp. 221–231, 2019, doi: 10.22266/IJIES2019.0228.22.

[10] J. Ramkumar and R. Vadivel, "Improved frog leap inspired protocol (IFLIP) – for routing in cognitive radio ad hoc networks (CRAHN)," *World J. Eng.*, vol. 15, no. 2, pp. 306–311, 2018, doi: 10.1108/WJE-08-2017-0260.

[11] J. Ramkumar and R. Vadivel, "CSIP—cuckoo search inspired protocol for routing in cognitive radio ad hoc networks," in *Advances in Intelligent Systems and Computing*, 2017, vol. 556, pp. 145–153. doi: 10.1007/978-981-10-3874-7_14.

[12] L. Mani, S. Arumugam, and R. Jaganathan, "Performance Enhancement of Wireless Sensor Network Using Feisty Particle Swarm Optimization Protocol," *ACM Int. Conf. Proceeding Ser.*, pp. 1–5, Dec. 2022, doi: 10.1145/3590837.3590907.

[13] D. Jayaraj, J. Ramkumar, M. Lingaraj, and B. Sureshkumar, "AFSORP: Adaptive Fish Swarm Optimization-Based Routing Protocol for Mobility Enabled Wireless Sensor Network," *Int. J. Comput. Networks Appl.*, vol. 10, no. 1, pp. 119–129, Jan. 2023, doi: 10.22247/ijcna/2023/218516.

[14] J. Ramkumar, C. Kumuthini, B. Narasimhan, and S. Boopalan, "Energy Consumption Minimization in Cognitive Radio Mobile Ad-Hoc Networks using Enriched Ad-hoc On-demand Distance Vector Protocol," *2022 Int. Conf. Adv. Comput. Technol. Appl. ICACTA 2022*, pp. 1–6, Mar. 2022, doi: 10.1109/ICACTA54488.2022.9752899.

[15] P. Menakadevi and J. Ramkumar, "Robust Optimization Based Extreme Learning Machine for Sentiment Analysis in Big Data," *2022 Int. Conf. Adv. Comput. Technol. Appl. ICACTA 2022*, pp. 1–5, Mar. 2022, doi: 10.1109/ICACTA54488.2022.9753203.

[16] J. Ramkumar, R. Vadivel, and B. Narasimhan, "Constrained Cuckoo Search Optimization

Based Protocol for Routing in Cloud Network," *Int. J. Comput. Networks Appl.*, vol. 8, no. 6, pp. 795–803, 2021, doi: 10.22247/ijcna/2021/210727.

[17] J. Ramkumar and R. Vadivel, "Multi-Adaptive Routing Protocol for Internet of Things based Ad-hoc Networks," *Wirel. Pers. Commun.*, vol. 120, no. 2, pp. 887–909, Apr. 2021, doi: 10.1007/s11277-021-08495-z.

[18] J. Ramkumar and R. Vadivel, "Whale optimization routing protocol for minimizing energy consumption in cognitive radio wireless sensor network," *Int. J. Comput. Networks Appl.*, vol. 8, no. 4, pp. 455–464, 2021, doi: 10.22247/ijcna/2021/209711.

[19] R. Jaganathan and R. Vadivel, "Intelligent Fish Swarm Inspired Protocol (IFSIP) for Dynamic Ideal Routing in Cognitive Radio Ad-Hoc Networks," *Int. J. Comput. Digit. Syst.*, vol. 10, no. 1, pp. 1063–1074, 2021, doi: 10.12785/ijcds/100196.

[20] J. Ramkumar and R. Vadivel, "Improved Wolf prey inspired protocol for routing in cognitive radio Ad Hoc networks," *Int. J. Comput. Networks Appl.*, vol. 7, no. 5, pp. 126–136, 2020, doi: 10.22247/ijcna/2020/202977.

[21] J. Ramkumar, K. S. Jeen Marseline, and D. R. Medhunhashini, "Relentless Firefly Optimization-Based Routing Protocol (RFORP) for Securing Fintech Data in IoT-Based Ad-Hoc Networks," *Int. J. Comput. Networks Appl.*, vol. 10, no. 4, p. 668, Aug. 2023, doi: 10.22247/ijcna/2023/223319.

[22] R. Chandrakar, R. Raja, R. Miri, U. Sinha, A. Kumar Singh Kushwaha, and H. Raja, "Enhanced the moving object detection and object tracking for traffic surveillance using RBF-FDLNN and CBF algorithm," *Expert Syst. Appl.*, vol. 191, p. 116306, 2022, doi: 10.1016/j.eswa.2021.116306.

[23] Q. Zhang, X. Hu, Y. Yue, Y. Gu, and Y. Sun, "Multi-object detection at night for traffic investigations based on improved SSD framework," *Heliyon*, vol. 8, no. 11, p. e11570, 2022, doi: 10.1016/j.heliyon.2022.e11570.

[24] I. García-Aguilar, J. García-González, R. M. Luque-Baena, and E. López-Rubio, "Automated labeling of training data for improved object detection in traffic videos by fine-tuned deep convolutional neural networks," *Pattern Recognit. Lett.*, vol. 167, pp. 45–52, 2023, doi: 10.1016/j.patrec.2023.01.015.

[25] K. Pawar and V. Attar, "Deep learning based detection and localization of road accidents from traffic surveillance videos," *ICT Express*, vol. 8, no. 3, pp. 379–387, 2022, doi: 10.1016/j.icte.2021.11.004.

[26] H. (Frank) Yang, J. Cai, C. Liu, R. Ke, and Y. Wang, "Cooperative multi-camera vehicle tracking and traffic surveillance with edge artificial intelligence and representation learning," *Transp. Res. Part C Emerg. Technol.*, vol. 148, p. 103982, 2023, doi: 10.1016/j.trc.2022.103982.

[27] J. Qu, R. W. Liu, Y. Guo, Y. Lu, J. Su, and P. Li, "Improving maritime traffic surveillance in inland waterways using the robust fusion of AIS and visual data," *Ocean Eng.*, vol. 275, p. 114198, 2023, doi: 10.1016/j.oceaneng.2023.114198.

[28] Y. Xia, Z. Sun, A. Tok, and S. Ritchie, "A dense background representation method for traffic surveillance based on roadside LiDAR," *Opt. Lasers Eng.*, vol. 152, p. 106982, 2022, doi: 10.1016/j.optlaseng.2022.106982.

[29] A. Pramanik, S. Sarkar, and S. K. Pal, "Video surveillance-based fall detection system using object-level feature thresholding and Z−numbers," *Knowledge-Based Syst.*, vol. 280, p. 110992, 2023, doi: 10.1016/j.knosys.2023.110992.

[30] A. Pramanik, S. Sarkar, and J. Maiti, "A real-time video surveillance system for traffic pre-events detection," *Accid. Anal. Prev.*, vol. 154, p. 106019, 2021, doi: 10.1016/j.aap.2021.106019.

[31] J. J. Park, K. A. Park, T. S. Kim, S. Oh, and M. Lee, "Aerial hyperspectral remote sensing detection for maritime search and surveillance of floating small objects," *Adv. Sp. Res.*, vol. 72, no. 6, pp. 2118–2136, 2023, doi: 10.1016/j.asr.2023.06.055.

[32] M. Rezaei, M. Azarmi, and F. M. P. Mir, "3D-Net: Monocular 3D object recognition for traffic monitoring," *Expert Syst. Appl.*, vol. 227, p. 120253, 2023, doi: 10.1016/j.eswa.2023.120253.

[33] S. Yuan, Q. Zhang, L. Zhu, S. Wang, Y. Zang, and X. Zhao, "Multi-level object detection by multi-sensor perception of traffic scenes," *Neurocomputing*, vol. 514, pp. 486–499, 2022, doi: 10.1016/j.neucom.2022.09.020.

[34] S. Alam Ansari and A. Zafar, "A fusion of dolphin swarm optimization and improved sine cosine algorithm for automatic detection

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

and classification of objects from surveillance videos," *Meas. J. Int. Meas. Confed.*, vol. 192, p. 110921, 2022, doi: 10.1016/j.measurement.2022.110921.

[35] M. J. Akhtar *et al.*, "A Robust Framework for Object Detection in a Traffic Surveillance System," *Electronics (Switzerland)*, vol. 11, no. 21. 2022. doi: 10.3390/electronics11213425.

[36] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:1010933404324.

[37] B. Wang, X. P. Jin, and B. Cheng, "Lion pride optimizer: An optimization algorithm inspired by lion pride behavior," *Sci. China Inf. Sci.*, vol. 55, no. 10, pp. 2369–2389, Oct. 2012, doi: 10.1007/s11432-012-4548-0.